



UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
09/895,287	06/29/2001	Hanumantha Rao Susarla	5181-91701	7187

7590 04/13/2004

Robert C. Kowert
Conley, rosse, & Tayon, P.C.
P.O. Box 398
Austin, TX 78767

EXAMINER

VO, TED T

ART UNIT	PAPER NUMBER
----------	--------------

2122

DATE MAILED: 04/13/2004

3

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary

Application No.

09/895,287

Applicant(s)

SUSARLA ET AL.

Examiner

Ted T. Vo

Art Unit

2122

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If the period for reply specified above is less than thirty (30) days, a reply within the statutory minimum of thirty (30) days will be considered timely.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

- 1) ☒ Responsive to communication(s) filed on 29 June 2001.
- 2a) ☐ This action is FINAL. 2b) ☒ This action is non-final.
- 3) ☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

- 4) ☒ Claim(s) 1-62 is/are pending in the application.
- 4a) Of the above claim(s) _____ is/are withdrawn from consideration.
- 5) ☐ Claim(s) _____ is/are allowed.
- 6) ☒ Claim(s) 1-62 is/are rejected.
- 7) ☐ Claim(s) _____ is/are objected to.
- 8) ☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

- 9) ☐ The specification is objected to by the Examiner.
- 10) ☐ The drawing(s) filed on _____ is/are: a) ☐ accepted or b) ☐ objected to by the Examiner.
Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).
Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).
- 11) ☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

- 12) ☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).
- a) ☐ All b) ☐ Some * c) ☐ None of:
1. ☐ Certified copies of the priority documents have been received.
2. ☐ Certified copies of the priority documents have been received in Application No. _____.
3. ☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

- 1) ☒ Notice of References Cited (PTO-892)
- 2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
- 3) ☐ Information Disclosure Statement(s) (PTO-1449 or PTO/SB/08)
Paper No(s)/Mail Date _____.
- 4) ☐ Interview Summary (PTO-413)
Paper No(s)/Mail Date. _____.
- 5) ☐ Notice of Informal Patent Application (PTO-152)
- 6) ☐ Other: _____.

DETAILED ACTION

1. This action is in response to the communication filed on 6/29/2001.

Claims 1-62 are pending in the application.

Claim Rejections - 35 USC § 112

2. The following is a quotation of the second paragraph of 35 U.S.C. 112:
The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

Claims 11, 19, 24, 28, 35, 51, and 60, are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

As per Claims 11, 19, 24, 28, 35, 51, and 60: Each claim contains the trademark/trade name: Java 2 Platform, Java 2 Enterprise Edition (J2EE), or/and Enterprise Java Bean. Where a trademark or trade name is used in a claim as a limitation to identify or describe a particular material or product, the claim does not comply with the requirements of 35 U.S.C. 112, second paragraph. See *Ex parte Simpson*, 218 USPQ 1020 (Bd. App. 1982). The claim scope is uncertain since the trademark or trade name cannot be used properly to identify any particular material or product. A trademark or trade name is used to identify a source of goods, and not the goods themselves. Thus, a trademark or trade name does not identify or describe the goods associated with the trademark or trade name. In the present case, the trademark/trade name is used to identify/describe a product, accordingly, the identification/description is indefinite.

Claim Rejections - 35 USC § 102

3. The following is a quotation of the appropriate paragraphs of 35 U.S.C. 102 that form the basis for the rejections under this section made in this Office action:

A person shall be entitled to a patent unless –

(b) the invention was patented or described in a printed publication in this or a foreign country or in public use or on sale in this country, more than one year prior to the date of application for patent in the United States.

4. Claims 1-62 are rejected under 35 U.S.C. 102(b) as being anticipated by Liang et al., "Dynamic Class Loading in Java Virtual Machine, 1998 ACM.

Given the broadest reasonable interpretation of followed claims in light of the specification.

As per Claim 1: Liang discloses:

A method for reloading classes in an application, the method comprising:

a class loader loading a class in the application, wherein the class loader is one of a hierarchal stack of class loaders each configured to load one or more classes in the application (See page 37, Figure 2, Section 2.1, Overview of Class Loading)

detecting the class has been changed (See Figure 3, page 39);

replacing the class loader in the hierarchy of class loaders with a new class loader for the detected changed class; and the new class loader reloading the changed class in the application (See page 39, left column, last full paragraph);

wherein said detecting the class has been changed, said replacing the class loader and said reloading the changed class are performed while the application is executing (See page 38, right column, second full paragraph, "defineClass method"; and see page 39, left column, first paragraph of section 3.1, "The upgrade must not require the application to shut down or restart").

As per Claim 2: Liang discloses, *"The method as recited in claim 1, wherein only the changed class and other classes with dependencies on the changed class are reloaded in response to said detecting the class has been changed"* (See page 39, left column, second paragraph in section 3.1, "subset");

As per Claim 3: Liang discloses, *"The method as recited in claim 1, wherein said detecting the class has been changed, said replacing the class loader and said reloading the changed class are performed without restarting the application"* (See page 39, left column, first paragraph in section 3.1, "restart").

As per Claim 4: Liang discloses, *"The method as recited in claim 1, further comprising: a class loader controller receiving a request to load the class prior to the class loader loading the class; the class loader controller determining that the class loader in the hierarchal stack of class loaders is responsible for loading the class; and the class loader controller invoking the class loader to perform said loading the class in the application"* (See Page 39, full right column).

As per Claim 5: Liang teaches, *"The method as recited in claim 1, further comprising: registering the loaded class with a dirty class monitor; and the dirty class monitor performing said detecting the class has been changed"*, inherently in Figure 3.

As per Claim 6: Liang teaches, *"The method as recited in claim 5, further comprising: the dirty class monitor notifying a class loader controller that the class has been changed; the class loader controller performing said replacing the class loader in the hierarchy of class loaders with the new class loader for the class in response to said notification; and the class loader controller invoking the new class loader to perform said reloading the changed class in the application"*, inherently in Figure 3.

As per Claim 7: Regarding, *"The method as recited in claim 1, further comprising: determining one or more classes with dependencies on the changed class; replacing one or more class loaders in the hierarchy of class loaders, wherein the one or more class loaders are each configured to load one or more of the one or more classes with dependencies on the changed class; and the one or more class loaders each reloading the one or more classes in the application with dependencies on the changed class which the particular class loader is configured to load, wherein said reloading is performed while the application is executing"*: (see page 38, right column, second paragraph, "delegation relationship between class loaders"). Furthermore, the claim has the functionality corresponding to the replace step recited in

claim 1, and it is inherent in Class hierarchy. Therefore, the rejection of this claim has the same reason as set for in the claim 1.

As per Claim 8: Liang discloses "The method as recited in claim 1, wherein the application is one of a plurality of applications executing within an application server (See page 39, Figure 3), wherein each of the one or more applications is associated with an application-specific hierarchy of class loaders configured to load classes in the particular application (see page 37, Figure 2).

As per Claim 9: Liang discloses, "The method as recited in claim 8, wherein the application-specific hierarchy of class loaders in each application is configured to load the classes in the particular application while the particular application is executing" (see page 37, Figure 2).

As per Claim 10: Liang discloses, "The method as recited in claim 8, wherein each of the class loaders in the application-specific hierarchy of class loaders in each application is configured to be replaced to reload one or more changed classes in the particular application while the particular application is executing" (see page 39, section 3.1).

As per Claim 11: Liang discloses, "The method as recited in claim 8, wherein the application server is based on the Java™ 2 Platform, Enterprise Edition (J2EE™)" because this limitation recites to a product that is incorporated by Liang.

As per Claim 12: Liang discloses, "*The method as recited in claim 1, wherein the application comprises one or more modules, wherein the hierarchical stack of class loaders includes a module class loader for each module in the application, and wherein the module class loader associated with a particular module is configured to load one or more classes of the particular module*" in teaching the Class Loaders, section 2 ("Java virtual machine" known as stack machine).

As per Claim 13: Liang discloses, "*The method as recited in claim 12, wherein the hierarchical stack of class loaders further includes an application class loader (See Figure 2, page 37, "application class loader, Browser code"), wherein the application class loader is the parent class loader of the one or more module class loaders in the hierarchical stack (Virtual Machine) of class loaders*" (See Figure 2, page 37).

As per Claims 14-18: The claims recites the class loaders that load correspondingly classes. See pages 37, section 2: "the purpose of Class loaders is to support dynamic loading of software components", and see section 2.1: Overview of Class Loading, and refer to Figure 2.

As per Claim 19: Liang discloses, "*The method as recited in claim 12, wherein the stack of class loaders further includes one or more Enterprise JavaBeans (EJB) class loaders, wherein each of the one or more EJBTM class loaders is a child of one module class loader in the hierarchical stack of class loaders*" because this limitation recites to a product that is incorporated by Liang.

As per Claim 20: Liang discloses, "*The method as recited in claim 12, wherein the stack of class loaders further includes one or more Web class loaders, wherein each of the one or more Web class loaders is a child of one module class loader in the hierarchical stack of class loaders*" (See Figure 2, "Class loaders in a web browser").

As per Claim 21: Regarding,

"A method for dynamically reloading classes in an application executing within an application server, the method comprising:

changing a class used by the application;

replacing a class loader for the class in the application with a new class loader for the changed class, wherein the class loader is one of a hierarchal stack of class loaders each configured to load one or more classes in the application;

replacing one or more class loaders for one or more classes with dependencies on the changed class, wherein the one or more class loaders are included in the hierarchical stack of class loaders;

the new class loader reloading the changed class; and

the replaced one or more class loaders reloading the one or more classes in the application with dependencies on the changed class; wherein only the class loaders for the changed class and the one or more classes with dependencies on the changed class are replaced in response to said changing the class; and wherein said replacing the class loaders and said reloading the classes are performed while the application is executing without restarting the application": The claim recites the limitation that has the

functionality corresponding to the limitation recited in Claim 1. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 1.

As per Claim 22: Regarding, "*The method as recited in claim 21, wherein the application comprises one or more modules, wherein each module in the application is associated with a module class loader for the particular module configured to load one or more classes of the particular module, and wherein the one or more module class loaders are included in the hierarchical stack of class loaders*": The claim recites the limitation that has the functionality corresponding to the limitation recited in Claim 12. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 12.

As per Claim 23: Regarding, "*The method as recited in claim 22, wherein the hierarchical stack of class loaders includes an application class loader configured to load classes used by more than one module in the application, and wherein the application class loader is the parent class loader of the one or more module class loaders in the hierarchical stack of class loaders*": The claim recites the limitation that has the functionality corresponding to the limitation recited in Claim 13. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 13.

As per Claim 24: Liang discloses, "*The method as recited in claim 23, wherein the hierarchical stack of class loaders further includes one or more Enterprise JavaBeans (EJB) class loaders, wherein each EJB.TM. class loader is a child of one module class loader in the hierarchical stack of class loaders*", because this limitation recites to a product that is incorporated by Liang.

As per Claim 25: Regarding, "*The method as recited in claim 23, wherein the hierarchical stack of class loaders further includes one or more Web class loaders, wherein each Web class loader is a child of one module class loader in the hierarchical stack of class loaders*": The claim recites the limitation that has the functionality corresponding to the limitation recited in Claim 20. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 20.

As per Claim 26: Regarding, "*The method as recited in claim 23, wherein the application server includes a system class loader configured to load core classes of the application server, wherein the system class loader is the parent class loader of the application class loader in the hierarchical stack of class loaders*":

The claim recites the limitation that has the functionality corresponding to the limitation recited in Claim 16, addressed in the rejection of Claims 14-18. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claims 14-18.

As per Claim 27: regarding, *"The method as recited in claim 21, wherein the application server is operable to execute a plurality of applications, wherein each application includes a hierarchical stack of class loaders configured to load classes for the particular application"* (Inherently in Figure 2, page 37).

As per Claim 28: Liang discloses, *"The method as recited in claim 21, wherein the application server is based on the Java™ 2 Platform, Enterprise Edition (J2EE™)"* because this limitation recites to a product that is incorporated by Liang.

As per claim 29:

"A system comprising: a processor; a memory operable to store program instructions, wherein the program instructions implement an application server executable by the processor within the system, wherein the program instructions further implement a plurality of applications executable by the processor within the system; wherein the application server is operable to provide access to the plurality of applications to clients of the application server; wherein one or more of the plurality of applications each includes a dynamic class reloading module comprising a hierarchical stack of class loaders, wherein the hierarchical stack of class loaders includes a separate class loader for each module in the particular application, and wherein each class loader is operable to reload one or more classes used by the particular application; wherein, for each of the one or more applications, the dynamic class reloading modules is operable during execution of the application to: detect that a class used by the application has been changed; replace a class loader for the class in the hierarchical stack of class loaders with a new class loader for the detected changed class; and wherein the new class loader is operable to reload the changed class in the first application during execution of the first application": The claim recites a system that has the claim functionality corresponding to the limitation recited in Claim 1. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 1.

As per claim 30: Regarding *"The system as recited in claim 29, wherein said detecting, said replacing and said reloading are performed without restarting the application"*: The claim recites a system that has the

Art Unit: 2122

claim functionality corresponding to the limitation recited in Claim 3. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 3.

As per claim 31: "The system as recited in claim 30, wherein said detecting, said replacing and said reloading are performed without restarting the application server": The claim recites the system that has the claim functionality corresponding to the limitation recited in Claim 3. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 3.

As per Claim 32: "The system as recited in claim 29, wherein, for each of the one or more applications, the dynamic class reloading modules is further operable during execution of the application to replace one or more other class loaders in response to said detecting, wherein the one or more other class loaders are operable to reload one or more classes with dependencies on the changed class": The claim recites a system that has the claim functionality corresponding to the limitation recited in Claim 7.

Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 7.

As per Claim 33: "The system as recited in claim 29, wherein each dynamic class reloading module further comprises a class loader controller operable to: receive a notification that the class has been changed; determine which class loader in the hierarchical stack of class loaders is operable to load the class; perform said replacing the class loader with the new class loader; and invoke the new class loader to perform said reloading the changed class": The claim recites a system that has the claim functionality corresponding to the limitation recited in Claim 4. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 4.

As per Claim 34: Regarding claim limitation, "*The system as recited in claim 33, wherein each dynamic class reloading module further comprises a dirty class monitor operable to: perform said detecting that the class used by the application has been changed; and notify the class loader controller that the class has been changed*": The claim recites a system that has the claim functionality corresponding to the limitation recited in Claim 6. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 6.

As per Claim 35: Liang discloses, "The system as recited in claim 29, wherein the application server is based on the Java™ 2 Platform, Enterprise Edition (J2EE™)" because this limitation recites to a product that is incorporated by Liang.

As per Claim 36:

"A system comprising:

a processor; a memory operable to store program instructions, wherein the program instructions implement an application executable by the processor within the system, wherein the application includes a dynamic class reloading module comprising a hierarchical stack of class loaders, wherein each of the hierarchical stack of class loaders is executable to load one or more classes in the application;

wherein the application is executable within the system to: invoke a class loader to load a class in the application, wherein the class loader is one of the hierarchal stack of class loaders; detect the class has been changed; replace the class loader in the hierarchy of class loaders with a new class loader for the detected changed class; and invoke the new class loader to reload the changed class in the application;

wherein said detecting the class has been changed, said replacing the class loader and said reloading the changed class are performed while the application is executing": The claim recites a system that has the claim functionality corresponding to the limitation recited in Claim 1. Therefore, the claim is rejected in the same reason as set forth in connecting to the rejection of Claim 1.

As per Claims 37-43: The claims recite a system that has the claim functionality corresponding to the limitations recited in Claims 2-7. Therefore, the claims are rejected in the same reasons as set forth in connecting to the rejections of Claims 2-7.

As per Claims 44-47: The claims recite a system that has the claim functionality corresponding to the limitations recited in Claims 8-11. Therefore, the claims are rejected in the same reasons as set forth in connecting to the rejections of Claims 8-11.

As per Claims 48-52: The claims recite a system that has the claim functionality corresponding to the limitations recited in Claims 12-13, 16, 19-20. Therefore, the claims are rejected in the same reasons as set forth in connecting to the rejections of Claims 12-13, 16, 19-20.

Art Unit: 2122

As per Claims 53-62: The claims recite a carrier medium that has the claim functionality corresponding to the limitations recited in Claims 1-5, 6-8, 11-13, and 16. Therefore, the claims are rejected in the same reasons as set forth in connecting to the rejections of Claims 1-5, 6-8, 11-13, and 16.

Conclusion

5. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Venkatraman et al., US 6,571,388 B1, discloses a method for customizing class loading.

Gong, "Secure Java Class Loading", 1998 IEEE, discloses a Java security model.

IBM Technical Disclosure Bulletin, "Java Dynamic Class Loader", 1996, discloses a method for selectively loading a class in Java.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ted T. Vo whose telephone number is (703) 308-9049. The examiner can normally be reached on Monday-Friday from 8:00 AM to 5:30 PM ET. If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Tuan Dam, can be reached on (703) 305-4552.

The fax phone numbers:

(703) 872-9306 (for formal communication intended for entry);

(703) 746-5429 (for informal or draft communication, please label "PROPOSED" or "DRAFT").

Any inquiry of a general nature or relating to the status of this application or proceeding should be directed to the Group receptionist whose telephone number is (703) 305-3900.

TED T. VO

Patent Examiner
Art Unit: 2122
April 8, 2004